

I Servlet Java

- Sono componenti java in esecuzione sul server in grado di interfacciarsi ad un server web
- Usano una tecnica di interfacciamento alternativa al CGI
- I package `javax.servlet` e `javax.servlet.http` (J2EE) offrono le classi necessarie per programmarli
- Riferimenti:

Differenze rispetto a CGI

- Un servlet, una volta creato e inizializzato, rimane attivo nella JVM del server (in un thread apposito)
- Ad ogni richiesta HTTP viene invocato un suo metodo
 - non c'è l'overhead legato all'avvio del programma e all'inizializzazione
 - esiste un ambiente di esecuzione che sopravvive alla singola interazione HTTP
- Il server web deve essere in grado di
 - fornire un ambiente di esecuzione java
 - chiamare il servlet in corrispondenza di determinate richieste

Ciclo di vita di un Servlet

- Il servlet viene creato automaticamente dal server ed inizializzato tramite il metodo `init()`
- Ad ogni richiesta, viene chiamato il metodo `service` (che, se non è ridefinito, chiama a sua volta `doGet()` o `doPost()`)
- Al termine della vita del servlet viene chiamato `destroy()`

Tipica Struttura di un Servlet

- Si estende la classe `HTTPServlet`
- Si ridefinisce `init()` per programmare le operazioni da eseguire una sola volta, alla creazione
- Si ridefinisce
 - `service()`, se si vogliono trattare tutte le richieste allo stesso modo
 - `doGet()` e `doPost()`, se si vogliono differenziare i due metodi
- Si ridefinisce `destroy()` per programmare eventuali operazioni conclusive

Il Metodo init()

```
public void init() throws ServletException;
```

- ServletException rappresenta generici errori interni al servlet.
 - Esempio: l'impossibilità a connettersi ad un database.
- Se il metodo init fallisce, il servlet viene distrutto e ricreato

Metodi di Servizio

```
public void service(HttpServletRequest req,  
                    HttpServletResponse res)  
    throws ServletException, IOException;
```

- req e res permettono di agire su richiesta e risposta HTTP
- doGet() e doPost() hanno prototipi analoghi

Interfacce verso la Richiesta e la Risposta

- Danno accesso a librerie di operazioni di codifica e decodifica.
- Esempi di operazioni sulla Richiesta:
 - `String getParameter(String name)`
 - `String getHeader(String name)`
 - `String getMethod()`
 - `String getQueryString()`
- Esempi di operazioni sulla Risposta:
 - `void setHeader()`
 - `void getWriter()`

Accesso al Contesto

- Un Servlet può accedere ad elementi del proprio contesto di esecuzione
 - Riferimenti ad altri servlet
 - Attributi di contesto
- L'accesso avviene attraverso un'interfaccia ServletContext:

```
ServletContext context =  
    this.getServletConfig().getServletContext();  
context.getServlet("OtherServlet");
```

Sessioni

- Viene fornito un supporto completo alle sessioni tramite la classe `HTTPSession`:
 - Generazione di sessioni, con memorizzazione di un id sul client (tramite cookie)
 - Recupero della sessione associata ad una risposta
 - Associazione di informazioni alla sessione
 - Cancellazione automatica di una sessione dopo un periodo di inattività

Esempio: Contare le Richieste

```
public void doGet(HttpServletRequest req,
                  HttpServletResponse res)
    throws ServletException, IOException {
    HttpSession session =
        req.getSession(true) // ricava o crea la sessione
    // ricava il valore associato a "mycounter"
    Integer ival = (Integer) session.getValue("mycounter");
    if (ival==null) // se non esiste
        ival = new Integer(1); // ne crea uno nuovo
    else // altrimenti
        ival = new Integer(ival.intValue()+1); // lo incrementa
    // alla fine il nuovo valore viene salvato
    session.putValue("mycounter", ival);
}
```